30 YEARS OF ABSTRACT INTERPRETATION: A DANISH PERSPECTIVE (by Neil Jones)

– many researchers –

DIKU, University of Copenhagen

DAIMI, University of Aarhus

IMM, Denmark’s Technical University

Roskilde University

Aalborg University
IDEA OF THIS WORKSHOP (as stated)

to bring together the research community of abstract interpretation, in con-
nection with POPL08, VMCAI08 and PEPM08, to

bring together the research community of abstract interpretation,

The invited speakers are intended to

cover the global spread of abstract interpretation worldwide during
the last 30 years,

including future trends, applications . . . in industry. Each speaker will present

the state of the art from his personal and geographical perspective

and show the major challenges and open problems that abstract interpreta-
tion has to face in the next future.

Panel discussions are intended to

bring together the community and fix the agenda

for the next years.
Well, ... compilers from 1950s through 1970s (and many even now) are *ugly*:

- Filled with dirty tricks
- Mysterious algorithms and terms
  - kill and gen sets, program snapshots,
  - unclear relations between program analysis results and behavior (or even correctness) of transformed programs
  - Unclear semantics:
    Different compilers (and even optimization levels) treat the same code in different ways.
- Much of this changed (at least in principle, and somewhat in practice) after the Cousots’ seminal paper first saw the light of day in 1977
A true cornucopia of results/ideas/papers/methods/theories/... were inspired by the famous POPL paper of 1977.

It was the first properly semantics-based approach to program analysis.

Some topics it inspired at DIKU:

- Abstract interpretation
- Flow analysis, program analysis
- Strictness analysis
- Partial evaluation
- Binding-time analysis
- Termination Analysis
A BOOK SHOWING EARLY INFLUENCE BY THE COUSOTS


- A chapter written by Patrick
- A chapter by the editors, inspired by the Cousot approach: Flow analysis and optimization of Lisp-like structures.
- Another chapter, ditto: Complexity of flow analysis, inductive assertion synthesis, and a language due to Dijkstra.
  - Identified *relational* versus *independent attribute* flow analyses
- Many more chapters, some by people in this room!
MORE HIGH POINTS

- **1981 workshop in Aarhus** on abstract interpretation and related topics (Cousots, Mycroft, Nielsons, ...)
  
- **1985 Copenhagen conference** on Programs as Data Objects (eds. H Ganzinger and N D Jones), 1986.
  - several papers on: *strictness analysis by abstract interpretation*

- **Canterbury meeting** (was it 1986?) on abstract interpretation, Abramsky and Hankin (eds.), many Danish participants

- **EU BRAs** (basic research actions) *Semantique* and *Atlantique* and *Daedalus*, Danish partners

- **Danish research projects also:** *Aarhus, Copenhagen, Roskilde*, Denmark’s Technical University and Aalborg University Center

- **A long series of Static Analysis Seminars**, many Danish participants
SOME DIKU-BASED WORK DIRECTLY ON ABSTRACT INTERPRETATION

- N D Jones and H Søndergaard. A Semantics-Based Framework for the Abstract Interpretation of Prolog. 1987
- M Rosendahl. Abstract Interpretation and Attribute Grammars. 1991
- T P Jensen. Abstract Interpretation in Logical Form. 1992
- M Rosendahl. Higher-Order Chaotic Iteration Sequences. 1993
- K Marriot, H Søndergaard and N D Jones. Denotational Abstract Interpretation of Logic Programs. 1994
- N D Jones and F Nielson. Abstract Interpretation: a Semantics-Based Tool for Program Analysis. 1994 (an influential survey paper)
SOME OF FLEMMING NIELSON’S RESEARCH DIRECTIONS IN ABSTRACT INTERPRETATION

▶ Denotational Abstract Interpretation (1979 - 1991)

- two-level denotational semantics
- theoretical bases on how to: prove analyses correct, systematically develop analyses, efficiently compute fixed points
- used tensor products to generalise relational analysis methods
- formal foundation to prove correctness of code generation


- introduction to abstract interpretation
- focusing on the Cousots’ widenings and narrowings
- and abstraction and concretization functions
- “entire book, including chap. on type and effect systems, was written according to the underlying philosophies of abstract interpretation.”
- “Notions like principal types and principal typings are clearly expressed in the terminology of Moore families”. (a Cousot cornerstone)

N D. Jones and S. Muchnick. A flexible approach to interprocedural data flow analysis. 1982

N D. Jones and Alan Mycroft. A relational approach to program flow analysis. 1986


MORE WORK DONE AT DIKU IN THIS DIRECTION

▶ T Reps, M Sagiv and S Horwitz. Interprocedural dataflow analysis via graph reachability. 1994

▶ T Reps, M Sagiv and S Horwitz. Interprocedural dataflow analysis via graph reachability. 1994


FLOW AND PROGRAM ANALYSIS, STRICTNESS ANALYSIS

- P Sestoft. Analysis and efficient implementation of functional programs. 1991
- C Gomard and P Sestoft. Path Analysis for Lazy Data Structures. 1992
- M H Sørensen. A Grammar-Based Data-Flow Analysis to Stop Deforestation. 1994

Strictness analysis

- T P Jensen. Context Analysis of Functional Programs. 1990
- M Rosendahl. Strictness analysis for attribute grammars. 1992
- F Henglein. Iterative Fixed Point Computation for Type-Based Strictness Analysis. 1994
Relevance: Neil first thought of partial evaluation as

- an intensional variant of constant propagation

- A key breakthrough to self-application was binding-time analysis. Guess how it was done: by an abstract interpretation on domain \{early, late\}.

A book and a few papers (there are many more . . .):


- T Æ Mogensen. Binding Time Analysis for Polymorphically Typed Higher Order Languages. 1989

- K Malmkjær. On Static Properties of Specialized Programs. 1991

- C Mossin. Polymorphic Binding Time Analysis. 1993

In essence: analysis of flow information not at one state but information flow concerning state transitions. [Yes, the Cousots were the first to talk about abstracting transition analysis!]


▶ C S Lee, N D Jones and A M Ben-Amram. The Size-Change Principle for Program Termination. 2001

▶ C S Lee. Program termination analysis in polynomial time. 2002

▶ C S Lee. Finiteness analysis in polynomial time. 2002

▶ D Sereni and N D Jones. Termination Analysis of Higher-Order Functional Programs. 2005

▶ J Avery. Size-change Termination and Bound Analysis. 2006

THANK YOU, PATRICK AND RADHIA, for

▶ seminal works way back when;
▶ putting program analysis on proper semantic foundations;
▶ continuing to inspire research in this many-faceted and fascinating field;
▶ sardonic but insightful and constructive comments at innumerable conferences;
▶ Bigger and broader: for creating an intellectual environment and playground with
  ● challenging problems
  ● problems that are both interesting and worth solving
  ● evolving a field with deep applicability: one that is not just another bubble of self-contained and ultimately irrelevant academic studies
▶ For creating something that would not have existed without you:

Thirty Years of Abstract Interpretation